Toward Visual Autonomous Ship Board Landing of a VTOL UAV

Jose Luis Sanchez-Lopez , Srikanth Saripalli , Pascual Campoy ,
Jesus Pestana and Changhong Fu

I. Introduction

In recent years, considerable resources have been devoted to the design, development and operation of Unmanned Aerial Vehicles (UAVs). The applications of such UAVs are diverse, ranging from scientific exploration and data collection, to provision of commercial services, military reconnaissance, and intelligence gathering. Other areas include law enforcement, search and rescue, and even entertainment. UAVs, particularly ones with vertical take-off and landing capabilities (VTOL), enable difficult tasks without endangering the life of human pilots. This potentially results in cost and size savings as well as increased operational capabilities and performance. Currently the capabilities of such UAVs are limited. A helicopter is a compact VTOL capable platform extremely manoeuvrable.

The autonomous landing of VTOL UAVs is a very important capability for autonomous systems, that would be very useful for various tasks as search and rescue, law enforcement, and military scenarios. Our challenge is to provide the UAV with the capability of autonomously land on ship deck platforms in extreme weather conditions.

Autonomous landing on a fixed platform is a problem that has been studied since the emergence of UAVs and some existing solutions are provided [12], [20].

Some approaches have been given for some authors in the autonomous landing on moving targets field, in only two degrees of freedom platforms [11], or simple moving platforms [17], [18].

However, autonomously landing on a ship deck platform continues to be studied, and has only recently been solved for very favourable weather conditions [19], [7], [1], [4].

Regarding the use of computer vision as the main sensor in UAVs, we can also find out some works [12], [11], [13], [17], [18], [20] but no one provides the six degrees of freedom 3D pose of the landing platform, computing usually the x-y coordinates and the angle (three degrees of freedom). Other authors use special helipads or landmarks [21], [22], [23].

Regarding the ship simulation, the movement of a ship at Sea is due to the effect of wave motion. The typical environmental conditions attributed to waves are grouped into several Sea States [1] (see figure 1).

Sea State	*World Meteorological Organisation*	
	Description	*Significant Wave Height (m)*
0	Calm (glassy)	0
1	Calm (ripples)	0–0.1
2	Smooth (wavelets)	0.1–0.5
3	Slight	0.5–1.25
4	Moderate	1.25–2.5
5	Rough	2.5–4.0
6	Very Rough	4.0–6.0
7	High	6.0–9.0
8	Very High	9.0–14.0
9	Phenomenal	Over 14

Fig. 1. Sea State Parameters. The Significant Wave Height is defined as the average value of the height (vertical distance between trough and crest) of the largest 1/3 of the waves present. The waves are modelled as sinusoidal function.

The ship can be modelled as a rigid body moving in the sea with six degrees of freedom, figure 2 (see [2]). Its movement in the sea depends on the Sea State, the physical parameters of the ship, and the wave direction.

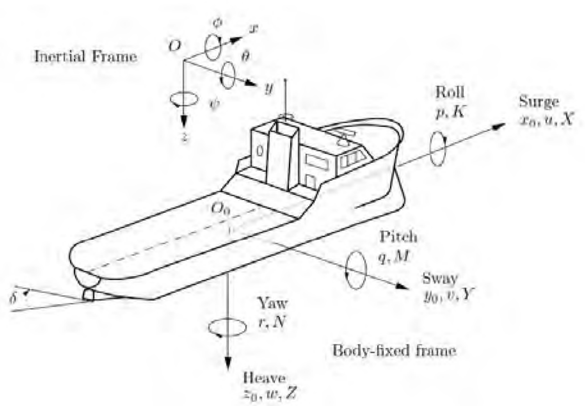

Fig. 2. Standard notation and sign conventions for ship motion description.

Some authors use sinusoidal functions with a fixed amplitude and frequency only for heave movement [3]. Others use sinusoidal functions for every degree of freedom [4]. Finally, others define a different heave movement function [5]. None of these authors take into account the ship model or sea state.

If we want to simulate the ship model, we can use physical models like the one used in games [6]; or control applications [2] or [7]. Unfortunately, these models are too complex, not realistic enough, and need additional program parameters.

To consider the ship model in the simplest form, a register of sailing data [8] could be used to calculate the model [9]. With this approach, the sea state and wave direction are ignored.

A better approach is the use of a simple physical model that consists of a sinusoidal function for each degree of freedom, whose parameters depend on the sea state, wave direction, and of course the ship [1]. This approach is not random enough for our problem because each degree of freedom moves periodically.

The paper is organized as follows: in section I a system and equipment description is developed; the ship deck simulation is explained in section III; section IV describes how the computer vision system works; the state estimator is explained in section V; results of the whole system are described in section VI; finally, section VII concludes the paper.

II. System Description

The VTOL UAV used in our study is a Rotomotion Inc, SR200 (figure 3). This helicopter is

Fig. 3. Rotomotion SR 200 Gas powered Helicopter. Length: 2790 mm; Width: 760 mm; Height: 860 mm; Main Rotor Diameter: 3000 mm; Endurance: Up to 5 hours; and Maximum Payload: 22.7 kg

equipped with an autopilot, an inertial measure unit (IMU), a GPS, and a small computer that simplifies the control task and is ideal for the development of autonomous capabilities for UAVs.

A white H surrounded by a white circle (figure 4) is painted on the heliport surface. These marks are the most extended marks to indicate the presence of a heliport surface.

Fig. 4. Helipad Marks used in our application. They are typical marks.

To detect the heliport, and to measure its pose with respect to the helicopter's pose, single downward looking colour camera computer vision system (a single camera with three channels: RGB). The selected camera is a Point Gray Inc, Chameleon USB colour camera (model CMLN-13S2C-CS), that can work with a resolution of 640 x 480 pixels at a rate of 24 frames per second (fps). We selected a single-camera system instead of a stereo pair because we assume that the size of the square landing platform is known. As such, the 3-D reconstruction could be calculated using the platform model and the camera calibration parameters (see section IV). The helicopter is also equipped with SONAR sensors that return the measurements of the distance to the floor when the helicopter is really close to it. These sensors allow us to detect the heliport pose in the very last stage of the landing when the helicopter is so near to the heliport that the computer vision system is not able to detect the marks.

III. Ship Deck Simulation

We simulate the movement of the ship deck on the Sea, using a Servos and Simulation Inc, Generic Motion System (model 710-6-500-220) with a $2.44 \times 2.44m^2$ gray surface as heliport (figure 5).

Our approach for the ship deck motion simulation is an improvement of [1]. We propose a uniform random generation for the amplitude

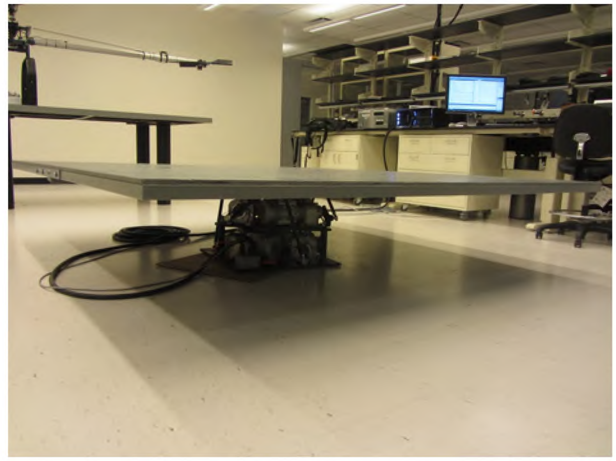

Fig. 5. Servos and Simulation Inc, 710-6-500-220 Generic Motion System. Number of axis: 6; Height: 48.6 cm; Floor Platform: $66 \times 68.6cm^2$; Power: 220 VAC @ 20 A; Payload: 226.8 kg; Max. Roll (x): $\pm 13°$; Max. Pitch (y): $\pm 15°$; Max. Yaw (z): $\pm 16°$; Max. Surge (x): $\pm 10.2cm$; Max. Sway (y): $\pm 10.2cm$; and Max. Heave (z): $\pm 6.4cm$

of each sinusoidal movement based on how the amplitude data corresponds to the top 1/10 waves. To obtain a continuous and derivable movement, we interpolate between two different sinusoidal function with a 5 degree polynomial.

Using MATLAB to achieve the ship simulation, we obtain, for a Sea State of 6, a Wave Direction of 60°, and a Oliver Hazard Perry Class FFG Frigate, the following plots (figures 6, 7 and 8). The shape of these plots look similar to the available plots of ship movements in [8].

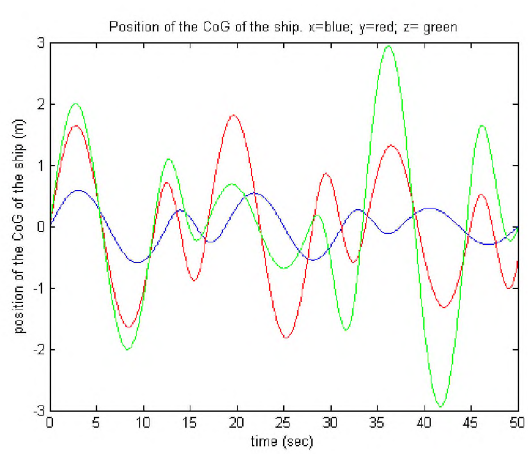

Fig. 6. Position of surge (x, blue), sway (y, red) and Heave (z, green) of the simulated ship's Center of Gravity.

Once the ship simulation is calculated, because

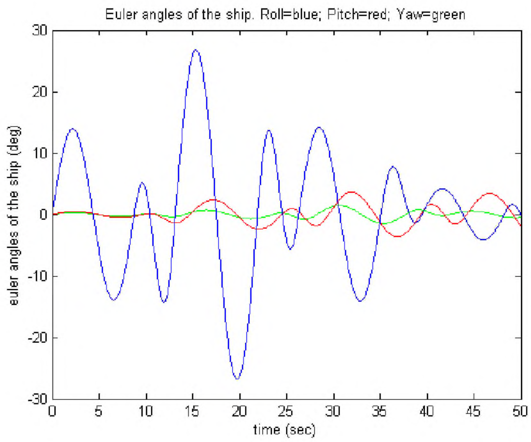

Fig. 7. Euler angles of the simulated ship: Roll (blue), Pitch (red) and Yaw (green).

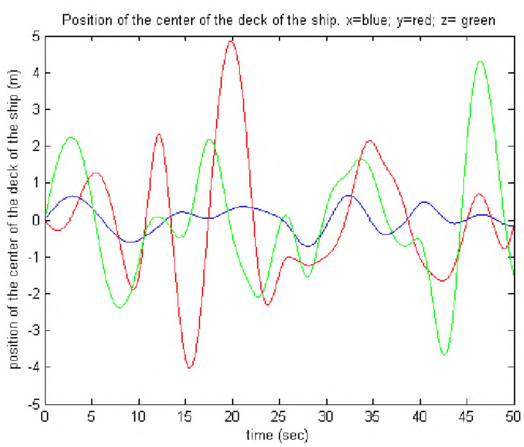

Fig. 8. Position of surge (x, blue), sway (y, red) and Heave (z, green) of the Center of the landing Deck of the simulated ship.

of our platform motion is smaller than the real ship's (figure 5), and our helicopter is smaller than its manned counterpart (figure 3), the entire system has to be scaled down. The approach that we choose for the scaling consist on scale only the amplitudes of the movement of each degree of freedom (DoF). We scale down the position DoF (x, y, and z) multiplying the simulated amplitude by the coefficient $1/90$, and the angles (Yaw, Pitch, and Roll) by $1/3$. While the scaling function is not realistic, it ensures that position and attitude are not being distorted and the platform is being used to the maximum extent possible.

The following step in the ship deck simulation is the calculation of the motor inputs of our platform through the Inverse Kinematics (figure 10) using the scaled ship simulation movement as the desired movement of our motion platform (figure 9). According to [10], we use the equation 1, defining a fixed reference system (attached to the bottom of the motion platform) and a mobile reference system (attached to the mobile part of the platform), to calculate the inverse kinematics:

$$L_i = \|O_r + O_{Rp} \cdot P_{bi} - O_{ai}\| \tag{1}$$

Where L_i is the longitude of the bar i of the motion platform; $O_r = [x, y, z]^t$ is the desired position of the mobile reference system with respect to the fixed one; O_{Rp} is the 3-by-3 rotation matrix of the desired attitude of the mobile reference system with respect to the fixed one; P_{bi} is the 3-by-1 vector of the position of the side of the bar i fixed to the mobile part of the platform, in coordinates of the mobile reference system; O_{ai} is the 3-by-1 vector of the position of the side of the bar i moved by the motor i, with respect to the fixed reference system. O_{ai} depends on the motor input q_i that is the unknown of the equation; and $i = 1..6$ indicates the number of the bar of the motion platform.

When equation 1 has no solution inside the compatible values of q_i, a singular configuration is achieved. If that happens, we calculate the value that minimizes equation 1 that is the nearest achievable pose by the platform with respect to the desired one.

The last step is the filtering of the calculated inputs in order to limit the speeds and accelerations because the inverse kinematics calculation does not take them into account (figure 11).

IV. Computer Vision System

In order to measure the pose of the Landing Platform, we use a single-camera computer vision system on board the helicopter as described in section II.

As the helipad (see section II) has no image descriptors (like SURF features) enough, the detection and the tracking cannot be based on matching them with a previously known template. We have to use other features of the helipad, like the colour or the marks (H surrounded by a circle).

The computer vision algorithm has the following steps:

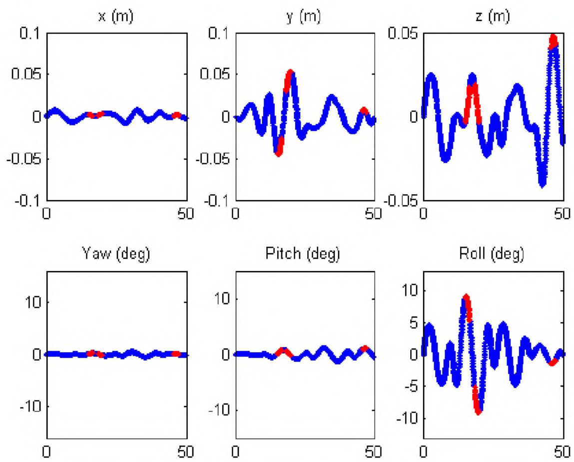

Fig. 9. Desired motion of our motion platform, for the ship simulation described in figures 6, 7 and 8. In red, points of singular configuration that are not achievable by our platform; in blue achievable points.

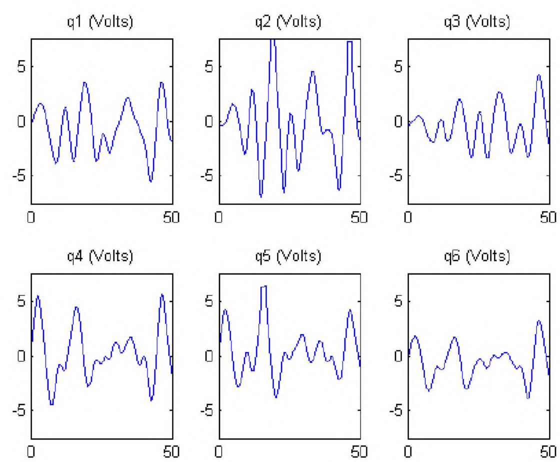

Fig. 10. Motor inputs (Volts) before filtering for the desired movement described in figure 9.

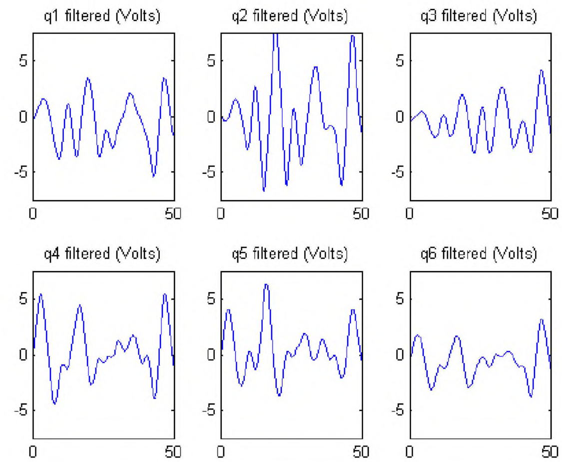

Fig. 11. Motor inputs (Volts) after filtering inputs in figure 10.

1) Image Acquisition and Preprocessing: section IV-A.
2) Heliport Zone Extraction: section IV-B.
3) Helipad Marks Extraction: section IV-C.
4) Heliport 3D Reconstruction: section IV-D.

The computer vision algorithm has been developed trying to maximize its accuracy, performance and robustness. We tried to avoid false positives using a very long decision tree. A false negative (no measure when it has to be measured) is better than a false positive (a wrong measurement), because the state estimator (section V) can manage it more easily.

A. Image Acquisition and Preprocessing

To start with the computer vision algorithm, the colour image is acquired (figure 12). The camera gives the image in the RGB (Red-Green-Blue) colour space. Then, the image is converted to an intensity image and to the HSV (Hue-Saturation-Value) colour space. Both images are preprocessed with a mean filter and then, with a opening morphological transformation. With this image preprocessing we are preparing the image to the following steps. If the preprocessing would not be done, the computer vision algorithm would work slower and with less accuracy, performance and robustness.

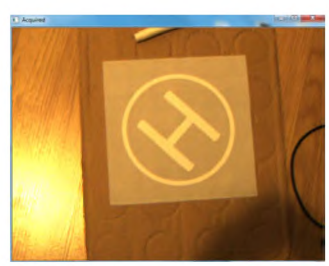

Fig. 12. Example of an Acquired Image

B. Heliport Zone Extraction

In this step, we work with the preprocessed HSV colour image. A colour thresholding is done to get a binary image with the grey pixels of the heliport. This binary image also requires a preprocessing step, that consist on a median filter followed by an opening morphological transformation. Then, the blobs are extracted and the small ones are deleted. With all these preprocessing, we clear all the noise and small regions. Finally, the blobs are filled in and, an OR logical transformation is done to get a whole binary image that represent the candidate pixels to belong to the heliport. Note that the heliport zone extraction give us not only the grey pixels, but also the white ones of the H and circle marks because of the blob's filled in that we did. Note also, that this step gives also other grey regions that can be visible in the image, giving us some false positives that we will filtered in the next stages.

Fig. 13. Example of an Intensity Image after Heliport Extraction

Fig. 14. Example of a Threshold Image, ready for look for Heliport Marks

C. Helipad Marks Extraction

To start with this stage, we element by element multiplication the intensity image obtained in the step described in section IV-A and the binary output image of section IV-B (figure 13). This resulting image is thresholded looking for the white pixels of the heliport marks (the H and the circle). In order to remove noise and prepare this binary image, it is preprocessed with a median filter and an opening morphological transformation. Then, the resulting blobs are calculated, filtering those with small area (figure 14). The segmentation steps ends, and the classification step starts.

The classification is done with a decision tree, where, each level gives some false positives, however, at the end of the tree, we will have one single solution. With this methodology, the speed and specially the accuracy of the computer vision algorithm is improved. In the first stages, an individual classification for Hs and circles is done. Then, we use both candidates (if found) to classify and verify them.

The first level is a fast classification using the Euler number (Euler number = connected components − number of holes) of each blob. H blobs have an Euler number equal to zero, and circle blobs' Euler number is one. Blobs with different Euler numbers are discarded. The Euler number is a scale, translation, rotation and homography invariant feature. The second level is a classification with a multi-layer perceptron (MLP) artificial neural network (ANN) with ten neurons in the hidden layer, tree outputs (H candidate, O candidate and other) and five inputs. The inputs are obtained with a principal component analyse (PCA) applied to the first seven invariant Hu Moments of each blob. Hu Moments are invariant to scaling, translation and rotation and are used in Optical Character Recognition (OCR), [16], [15], [14]. Homography modifies a little these features, but they can be used in our decision tree. As the H has more information than the circle (because it is less symmetrical that the circle), we can use it in the third classification level. This third level uses the signature of the H candidates. The signature is invariant to rotation and translation; it preserves its shape to scaling; and some features of the shape (relative maximum and minimum) are preserved to homography. The H signature has four relative maximum and four relative minimum. The four maximum are the external corners of the H; and the four minimum are the bisectrix of the horizontal segment (above and below the centroid), and the bisectrix of the vertical segments (the external points). Because of its symmetry, if we connect the four maximum, we have a quadrilateral polygon whose center should be near to the centroid of the

H blob. The same phenomenon appears with the minimum. The fourth level in our tree checks that all these three centers (center of maximum, center of minimum and centroid) are near. The fifth level checks the distance between the vertical straight lines of the maximum (vertical segment of the H) and the points of the minimum that should be in the vertical segment of the H. This distance has to be small (ideally zero).

Hitherto, our classification tree uses only individual features to achieve its task. Now, we have to select only one H blob and one circle blob among all the resulting candidates that have to be compatible both together. The sixth level is based on the knowledge that the H has to be inside the circle. The seventh and last level calculates the coefficient between the area of the H and the area of the circle, which should be, more or less, a constant value. In these two last levels all H and circle blob candidates are tested, discarding those that do not satisfy the conditions checked in the levels.

At the end of this step, we have the helipad marks (H and circle) extracted of the image (figures 15 and 16).

Fig. 15. Example of Circle selected blob

Fig. 16. Example of a H selected blob

D. Heliport 3D Reconstruction

The last step in the computer vision algorithm has to give us the 3D pose of the heliport with respect to the camera (on-board the helicopter). With the corners of the H of the helipad in the image (obtained thanks to its signature, see section IV-C), we can calculate the homography matrix between these points of the image and the same points in a predefined target image. Then, using the homography matrix, we calculate the corners of the heliport knowing where are the corners in the target image (figure 17).

Once we have the corners of the heliport in the image, the 3D reconstruction has to be performed (figure 18). The reconstruction is based on the pin-hole camera model (equations 2 and 3, been $i = 1..4$), the square and known platform model (equation 4, been $i, j = 1..4$ and $i \neq j$; and equation 5, been $i, j, k = 1..4$ and $i \neq j \neq k$). The camera has to be previously calibrated (focal distance f, scale factors K_x and K_y and principal point C_x and C_y, no distortion is assumed).

$$x_i \cdot f \cdot K_x - (x_{fi} - C_x) \cdot z_i = 0 \quad (2)$$

$$y_i \cdot f \cdot K_y - (y_{fi} - C_y) \cdot z_i = 0 \quad (3)$$

$$||\vec{x}_i - \vec{x}_j|| = L_{ij} \quad (4)$$

$$(\vec{x}_i - \vec{x}_j) \cdot (\vec{x}_j - \vec{x}_k) = 0 \quad (5)$$

Where $\vec{x}_i = [x_i, y_i, z_i]^t$ are the 3D coordinates of the point i in the central coordinate system, and x_{fi} and y_{fi} is the 2D coordinate of the point i in the camera lateral coordinate system.

Fig. 17. Example of Output Image after the computer vision algorithm. In green, the heliport. In red, the H corners (maximum of signature). In purple, the minimum of the H signature

V. State Estimation: Kalman Filter

In order to manage the measurements of the computer vision system, filtering the noise and

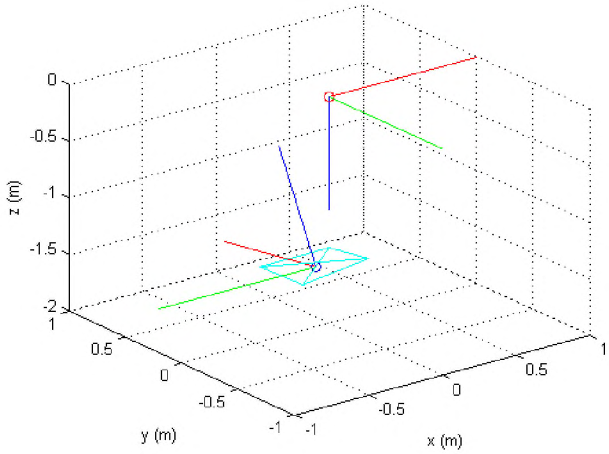

Fig. 18. Example of 3D reconstruction after the computer vision algorithm. The camera is fixed in the point (0, 0, 0), looking downwards

calculating the pose of the heliport even when measurements are not available, a state estimator is needed.

In our problem we can see three coordinate systems: the first one, the World frame, fixed to the ground; the helicopter frame, fixed to the camera on-board the helicopter; and the last one, the heliport frame, fixed to the landing platform. The movement of the helicopter with respect to the World is modelled thanks to the helicopter model, and can be measured thanks to the IMU and GPS. The movement of the landing platform with respect to the World is unpredictable and we have no measure of it, but we have the measure of the movement of the helicopter with respect to the landing platform (the output of the computer vision system). If we assume that we have a good estimation of the pose of the helicopter frame, we can transform the computer vision measure into a measure between the World frame and the landing platform. With this transformation, we decouple the models (but not the measures), and it is easier to define them.

As the movement of the ship deck platform with respect to the World frame is unpredictable, we cannot create any complex model to estimate its pose. We define the following easy model:

$$\frac{d^5 x_i}{dt^5} = 0 \tag{6}$$

Where x_i is the position of each DoF (x, y, z, θ, ψ, ϕ) of the landing platform.

With this model, an Extended Kalman Filter is implemented to obtain the pose of the heliport, using the computer vision measurements and knowing the state of the helicopter.

VI. Results

In this section, some examples of the results are shown. More results and videos can are available in the web of the first author (http://www.vision4uav.com/?q=jlsanchez/research).

A. Computer Vision System

In figures 19 to 21, the performance of the computer vision algorithm described in section IV is tested.

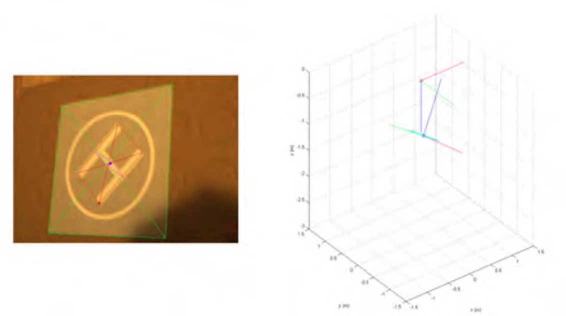

Fig. 19. Example of Heliport very tilted. The 3D reconstruction reflects the tilting, and the computer vision algorithm still works despite the huge tilting

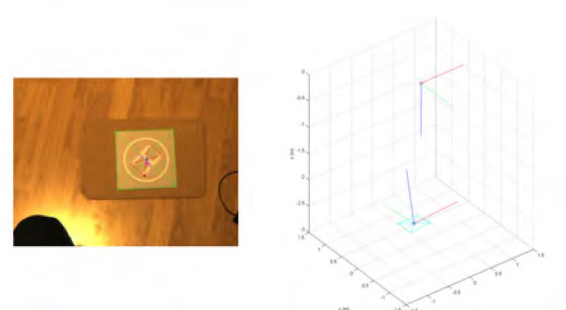

Fig. 20. Example of Heliport far away. The 3D reconstruction reflects the bigger distance. The computer vision algorithm is able to work in different ranges of heliport distances.

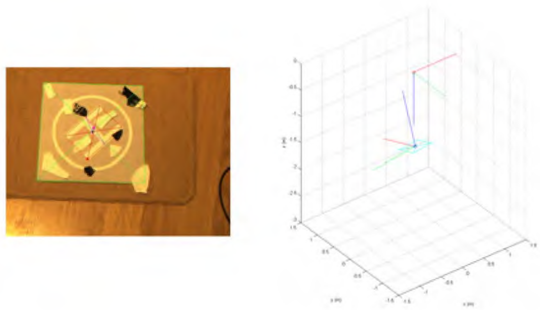

Fig. 21. Example of Contamination on the heliport. The computer vision algorithm works appropriately even with a really big and probably "unrealistic" contamination.

B. State Estimation

In figure 22, one of the DoF of the motion platform (the z movement) is estimated using the Kalman Filter with the model proposed in section V and the measures of the computer vision system (section IV) after the transformation to World frame's coordinates.

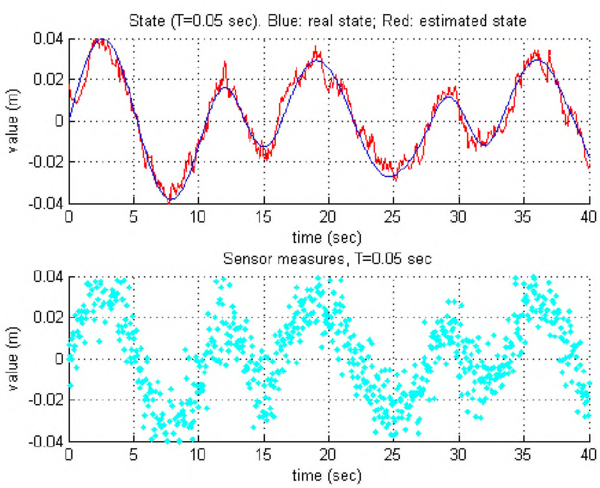

Fig. 22. Example of State Estimation of the pose of the motion platform. It is shown the z movement. In the upper plot, in blue, the real state, and in red, the estimated state. In the bottom plot, in cyan, the measurements of the computer vision system, with its typical noise, after the transformation to refer them to the World frame.

VII. Conclusion and Future Work

In this paper a new and complete ship deck simulation for the autonomous landing of VTOL UAVs on ships is proposed using a real Motion Platform. This simulation fulfils the requirements of being accurate, realistic, random and simple enough, therefore we can use it easily without losing realism. The pose of this landing platform is measured using a single-camera computer vision system on board the helicopter for standard grey helipads with an H surrounded by a circle. The computer vision requires the knowledge of the deck size for the 3D reconstruction. This algorithm was developed having in mind robustness, avoiding any false positive. Also, it works appropriately even with contamination on the helipad or light changes. A state estimator that uses the computer vision measures, calculates the state of the landing platform, removing its noise and avoiding the problems when the helipad is not detected. These are the first steps required to achieve a solution to the challenge of autonomously landing on a ship.

To complete this challenge, as future work, a state estimator which incorporates the helicopter model, and IMU measurements is required. Additionally, a controller will need to be designed and tested in order to close the control loop.

Acknowledgement

The authors would like to thank the Consejo Superior de Investigaciones Cientificas (CSIC) of Spain for the JAE-Predoctoral first author's scholarship and the UECMUAVS Project (PIRSES-GA-2010) included in the Marie Curie Program and the Spanish Ministry of Science MICYT DPI2010-20751-C02-01 for project funding.